Overview of Embedded Systems

Dept. of Electronics and Communication Engineering
Galgotias University

UNIT 1

- **UNIT I INTRODUCTION TO EMBEDDED SYSTEM**
- Basic components of Embedded system, Programming Language, Classification of Embedded system, Advantage & Disadvantage, Difference between Microprocessor & Microcontroller, Classification based on architecture, Memory Classification, Description of RAM, Description of CPU Registers, Introduction to Embedded C, Difference between C & Embedded C.

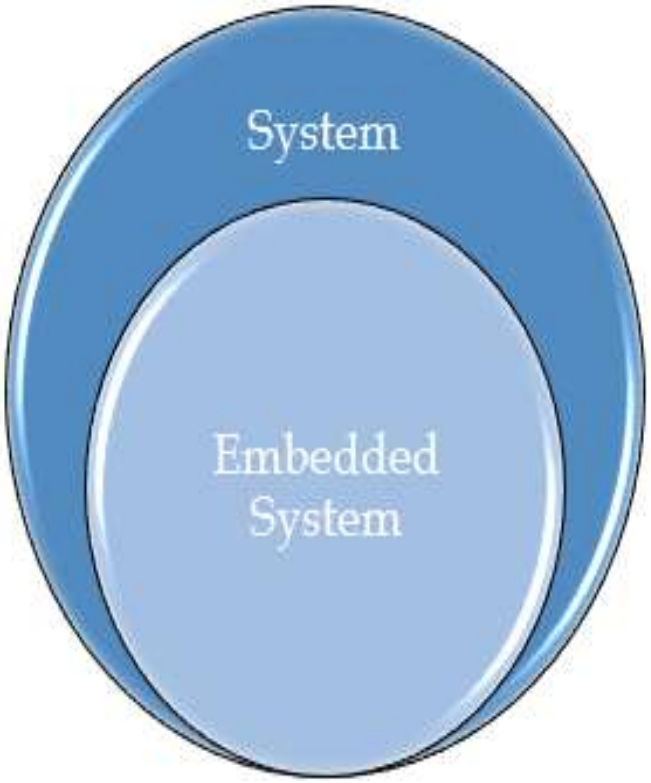
BOOKS

- **Text Books:**
- [T1] Design with PIC Microcontrollers, John B. Peatman, Pearson Education Asia, 2002
- [T2] ARM System Developer's Guide: Designing and Optimizing System Software, Andrew N. Sloss, Dominic Symes, Chris Wright, Morgan Kaufman Publication, 2004.
- [T3] Computers as components: Principles of Embedded Computing System Design, Wayne Wolf, Morgan Kaufman Publication, 2000

- **Reference Books:**
- [R1] The Design of Small-Scale embedded systems, Tim Wilmshurst, Palgrave2003
- [R2] Embedded System Design , Marwedel ,Peter , Kluwer Publishers , 2004.

WHAT IS SYSTEM ???

- A system is a group of units that are joined together to work in a specific routine and perform some fixed operation.
- These units could be of any nature i.e. if you are working on an electronics system then these units will be electronic components.
- Similarly, if you are working on some mechanical system then these units will be mechanical equipment or machinery etc.



SYSTEM EXAMPLES

WATCH

It is a time display SYSTEM

Parts: Hardware, Needles, Battery, Dial, Chassis and Strap

Rules

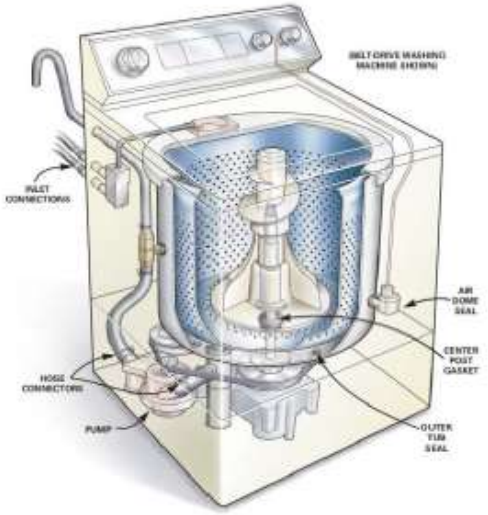
1. All needles move clockwise only
2. A thin needle rotates every second
3. A long needle rotates every minute
4. A short needle rotates every hour
5. All needles return to the original position after 12 hours

SYSTEM EXAMPLES

WASHING MACHINE

It is an automatic clothes washing SYSTEM

Parts: Status display panel, Switches & Dials, Motor, Power supply & control unit, Inner water level sensor and solenoid valve.



Rules

1. Wash by spinning
2. Rinse
3. Drying
4. Wash over by blinking
5. Each step display the process stage
6. In case interruption, execute only the remaining

WHAT IS EMBEDDED SYSTEMS ???

- **Embedded System - ** As its name suggests, Embedded means something that is attached to another thing. **Embedded System** is a computer **system embedded into an device, appliance** or a unit that is used to create any automation device or to control any machines using Micro Controller.

- Any sort of device which includes a programmable computer but itself is not intended to be a general-purpose computer”

- It includes hardware as well as software and it is a part of a larger system and is expected to function without human intervention.

- An embedded system is expected to respond, monitor as well as control external environment using sensors and actuators.

- So, basically what we are talking about is embedding a computer into an appliance and, that computer is not expected to be used for general purpose computing. Since it is embedded into an appliance, it needs to interact with the external world, so it has got analog interfaces.

Embedded Systems

- An embedded system is that system which has computer hardware with software embedded in it. Microcontrollers normally used to design embedded systems are Arduino, PIC Microcontroller, Atmel Microcontroller, 8051 Microcontroller etc.

- A general purpose computer such as Pentium PC is not an embedded system as it does not perform only specific functions. It can be used for a number of applications. Different programs can be installed on a PC for different tasks. A PC has a RAM and an operating system. The operating system loads application software into RAM and then the CPU runs it. While in the case of an embedded system, there is ROM in which application software is burned which can usually perform only one task.

- A PC itself is connected to many embedded systems, such as printer, keyboard, mouse, scanner, modem and many others. Such systems perform specific functions and have their own microcontrollers in them. I am going to explain this with some examples.

COMPUTER HARDWARE

A Microprocessor

A Large Memory
(Primary and Secondary)
(RAM, ROM and caches)

Input Units
(Keyboard, Mouse, Scanner, etc.)

Output Units
(Monitor, printer, etc.)

Networking Units
(Ethernet Card, Drivers, etc.)

I/O Units
(Modem, Fax cum Modem, etc.)

COMPONENTS OF EMBEDDED SYSTEM

- **It has Hardware**

  Processor, Timers, Interrupt controller, I/O Devices, Memories, Ports, etc.

- **It has main Application Software**

  Which may perform concurrently the series of tasks or multiple tasks.

- **It has Real Time Operating System (RTOS)**

  RTOS defines the way the system work. Which supervise the application software. It sets the rules during the execution of the application program. A small scale embedded system may not need an RTOS.

EXAMPLES OF EMBEDDED SYSTEM

- There's an endless list of Examples of Embedded Systems. Some of the most exciting Examples of Embedded Systems
  - Digital Camera
  - Automotive Embedded Systems
  - Home Security Systems
  - Automatic Washing Machine
  - Personal Digital Assistant
  - Industrial Robot
  - Automated Teller machine (ATM)
  - Calculator
  - Printer

Digital Camera

- A digital camera is very good example of embedded systems.
- Cameras that we use today are smart and have a lot of features that were not present in early cameras all because of embedded system used in them.
- A digital camera has basically three functions, to capture image which we call data, to store image data, and to represent this data.
- Today images are stored and processed in form of digital data in bits.
- There is no need of film for storing images. This feature has increased the storage capacity and made it easy to transfer images.
- In digital cameras, first image is captured and converted to digital form.
- This digital image is stored in internal memory.
- When the camera is attached to your personal computer for uploading images, it transfers the stored data.
- If I talk about smart camera, it has some extra features than digital cameras.

Digital Camera

- Smart cameras are able to capture details of the scene.
- These cameras analyze the images and are able to detect humans, motion, faces etc. from the whole image.
- For detection of objects in the image, some processing is required in cameras.
- Usually, image processing includes low level processing and high level processing.
- Various algorithms are available that are employed for this purpose.
- Components of a smart camera include,
  - image sensor that may be a CCD (Charge Coupled Device) or a CMOS (Complementary metal oxide semiconductor)
  - Analog to digital converter (A2D)
  - Image Processor
  - Memory
  - Lens
  - Led or other illuminating device
  - Communication Interface etc.
- Smart cameras may consist of some more devices depending on features.
- So, we can say that camera is one of the important embedded systems examples. It has its own processor, sensors, actuators and also memory for storage purposes

Automotive Embedded Systems

- Examples of embedded systems include automotive. Today cars use embedded systems replacing old traditional systems.
- Electronic Control Units are used in automotive embedded systems Examples.
- This unit contains microcontroller, switches, sensors, drivers, etc.
- All the sensors and actuators are connected to electronic control unit.
- Automobiles using embedded systems may consists of hundreds of microprocessors.
- Each microcontroller performs its own dedicated task. Some of them control engine. Some run dashboard devices.
- The whole system is actually comprised of several small systems.
- Using embedded systems in automotive has reduced the cost factor.
- It has improved the overall performance and increased functionality.
- It has also reduced weight and made automobile more safe and reliable.
- Applications of automotive embedded systems include:
  - Automatic Stability Control
  - Traction Control System
  - Pre-crash Safety System
  - Air bag
  - Car Navigation System
- So you can see that using embedded systems in automobiles is very useful and has increased the functionality of automotive.

Home Security System

- In embedded systems examples, I have another interesting one on my list that is home security system.
- Home security systems are used largely today.
- These systems have several features just as checking for fire or gas leakages, and detecting if someone suspicious tries to enter the house.
- A microcontroller is used for controlling all the operations. Sensors give data and if something wrong happens than safety alarms get activated.
- Sensors used in such systems include gas sensors, smoke sensors, temperature sensors, IR sensors etc.
- Also a keypad is included in such systems for entering password at the gate.
- If correct password is entered then this embedded system opens the gate and if someone tries to enter wrong password than alarm is set on and gates remain closed.
- The output is received from alarms or some display.
- The output can also be sent to some distant location.
- If family members are not present in home then still they can monitor the activities going on in their house.
- Home security system is not limited to homes.
- Such systems can be used at shops, stores and in industries.
- Almost every industry and office has security systems that can recognize the workers from their face or identity cards.
- Home automation system is also one of the examples of embedded systems as home security system.

Automatic Washing Machine

- Daily life examples of embedded systems include automatic washing machines and dryers.
- Washing clothes is not a difficult task now owing to embedded systems.
- You just have to add clothes and leave it to the machine. Rest operations are done by your machine itself.
- Machines have a microcontroller for controlling all the tasks.
- Sensors and actuators in this case are level sensors, valves, motor and also a display and keypad to input information.
- Once you load clothes in machine, the whole process consists of three cycles. Washing, rinsing and spinning. All the three cycles are initiated by machine itself. You just have to enter information for hot or cold water and press start button.
- During washing and rinsing cycle, water is added to the drum by pipes.
- Closing and opening of valves for adding water is checked through level sensors by microcontroller.
- Then the rotation of drum starts for pre-set time. After that water is drained out through pipes.
- During spinning cycle, water is not added and drum rotates for a set time.
- All the processes are controlled by microcontroller program.
- The timings for each cycle can be changed through the keypad.

Personal Digital Assistant

- Personal digital assistant (PDA) is the next example on my list. It's a device for personal use.
- You can find a lot of personal embedded systems examples like Mobile Phones, data organizers, PDA etc.
- Personal Digital Assistant is just like a personal computer in hand. It was used before smart phones came out.
- PDA is used as information manager and has the ability to connect to internet.
- It has a display mostly touch screen for the user to interact with the device.
- The display is used for entering data, memory card is used for storing data and it is provided with Bluetooth or WiFi for connectivity.
- Some of the personal digital assistants use keypads instead of touch screen to input information.
- This device is very handy in managing and sorting personal information. It is very light in weight and serves multiple functions.

Industrial Robots

- Embedded systems have a lot of applications in industries.
- Today, every process is being taken towards automation. So industrial robots are very important to mention with embedded systems examples.
- An industrial robot is an embedded system that comes in a variety of forms and each performs a number of different tasks. Some industrial robots are used for moving parts, tools, materials etc.
- Some are used in assembly operations while some of them are used in manufacturing.
- These robots have increased the productivity.
- They are widely used where precise operations are required or at the places which are difficult to access for humans.
- To understand the working of industrial robot as example of embedded systems, I am going to tell you about automated painting robots.
- Painting robots have a wide application area.
- They are replacing humans as they require less time for the whole operation and ensure best result.
- All of the activities are controlled through the program.
- Timing for the whole process and amount of paint is preset.
- Assembly robot is another example of industrial robots.
- The task of such robot is to create an assembly from all the parts.
- All parts are collected and assembled in correct sequence to form final product.
- There are a lot of examples of industrial robots.
- All are good embedded systems examples.

Automated Teller Machine

- An automated teller machine (ATM) is also an embedded system.
- It is a computerized device used in banking.
- You all are already familiar with its operation and use.
- A customer can access and perform his transactions without going to the bank and meeting some assistant.
- This machine consists of a card reader for detecting card and accessing information of the person.
- Also it has a keypad so the user can enter his commands and password.
- A screen displays information. A printer prints the receipts and cash is received from cash dispenser.
- A network is present between the bank computer and ATM machine through a host computer.
- All the data is verified with the bank computer and all transactions are stored in it.
- All these input and output operations are carried with the help of microcontroller.
- So this makes a one of the best examples of embedded systems.

Calculator

- Calculator is also one of the examples of embedded systems.
- Actually it is one of very earlier embedded system that is used widely.
- In this example, the function is to take input from the keypad, perform the required operation and show the results on LCD.
- Embedded Scientific Calculator has a high performance processor.
- A number of mathematical complex calculations can be performed by these calculators.
- You can also program such scientific calculators.
- With such kind of functionality, these calculators are very advance as compared to simple calculators, all because of embedded systems.

Printer

- A printer is an example of an embedded system.
- It does not need an external controller.
- It has its own control unit embedded in it
- The controller is programmed to perform only one function that is specific.
- The function performed by the printer is to read the data and print it on the paper.

FEATURES OR CHARACTERISTICS:

Single-functioned /Task Specific

- As I told in the previous section that an embedded system is not general purpose
- It is designed to perform special functions. For e.g. A pager always functions as a pager.
- So, embedded systems are designed for doing some specific tasks and then they are installed and those embedded systems keep on doing their specific tasks.

Sophisticated functionality

- An embedded system possesses sophisticated functionality. The degree of sophistication can vary from appliance to appliance.
- Embedded system being used in Microwave oven has simple functions so 16 bit microcontroller can be used in that while in automotive applications complex embedded system having 32 bit microcontrollers are used.

FEATURES OR CHARACTERISTICS:

Real Time Operation

- Another feature of these systems is that they work in real time.
- The basic definition is that operations must be completed by deadlines. So if I have a deadline, a real time operation must be completed within deadline.
- We have two kinds of real time deadlines- hard real time deadlines and soft real time headlines and accordingly also, we classify real time systems.
- In a hard real time systems, we cannot really miss a deadline. If you are talking about an atomic reactor control, if I miss a deadline then there can be a catastrophe.
- On the other hand, for a soft real time systems, we can at times miss deadline. See for example, when we are playing a video on a laptop even if we cannot decode a frame in time, nothing catastrophic happens, only it disturbs you viewing experience.

FEATURES OR CHARACTERISTICS

Tightly constrained

- All computing systems have constraints on design metrics, but those on an embedded system can be especially tight. Design metrics is a measure of an implementation's features such as its **cost, size, power, and performance**. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.

Performance

- The overall performance and accuracy of an embedded system also counts.
- It is measured considering all the conditions and constraints on the system.
- So, if your embedded system is just measuring the atmospheric temperature then it will be quite efficient because its quite an easy task.

Cost

- The cost factor is another important feature.
- Such systems are built for performing specific functions, and in large quantities.
- The design process is costly but once a system is designed, customized and produced in bulk, overall cost becomes minimum.

Size

- One of the feature of an embedded system is its size.
- The size should be small and it is done by adding more functionality in a single chip so that the need for external parts is reduced.

FEATURES OR CHARACTERISTICS

Power Consumption

- The power consumption is also low.
- This feature is becoming more and more important in new systems.
- Sometimes it happen that your embedded system has to be isolated and need to run for a very long time so in such cases the power consumption is a critical factor and it has to be really low.

Reactive and Reliability

- Embedded systems are reliable if they are operated under normal conditions. Embedded systems should be highly reliable and stable. Unlike usual computers, embedded systems use different underlying software that cannot be modified by consumers. Since they will be used for long periods of time and cannot be programmed easily, they're expected to run without any problems. In the case of some applications like undersea communication cables, navigation beacons or automobile components, maintenance is extremely difficult, if not highly improbable, so reliability is paramount.
- Embedded systems are usually feedback oriented or reactive. They function either by interacting with external stimuli, or depend on another system to give them an input.

FEATURES OR CHARACTERISTICS

Microprocessors based

- It must be microprocessor or microcontroller based.

Memory

- It must have a memory, as its software usually embeds in ROM. It does not need any secondary memories in the computer.

Connected

- It must have connected peripherals to connect input and output devices.

HW-SW systems

- Software is used for more features and flexibility. Hardware is used for performance and security.

CONSTRAINTS FOR AN EMBEDDED SYSTEM

The constraints on designing of almost every embedded system:

- Available System Memory
- Available processor speed
- Power Dissipation
- Size
- Cost

Components of embedded systems

- The components used to build up an embedded system can be categorized into four classes.
- **Analog Components**
- **Digital Components**
- **Converters**
- **RTOS and Software**

- **Analog Components**
- Analog Components are very necessary components of such a system. They are important because they help in interacting with real world. Examples of analog components are:
- Sensors.
- Actuators.
- Controllers.

Components of embedded systems

Digital Components

- Digital components mostly reside on the chip and do the processing operations. Examples are:
- Processors.
- Co-processors.
- Memory.
- Controllers.
- Buses.
- Application Specific Integrated Circuits (ASIC).

Converters

- Converters are very important building blocks. They are useful so that digital and analog components can work together. For example:
- Analog to Digital Converter (A2D).
- Digital to Analog Converter (D2A).

Components of embedded systems

- **Software**

  Without software an embedded system cannot work. The software is written in high level language. This software is burnt to some non volatile memory. Examples are:
- Application Programs.
- Exception Handlers.

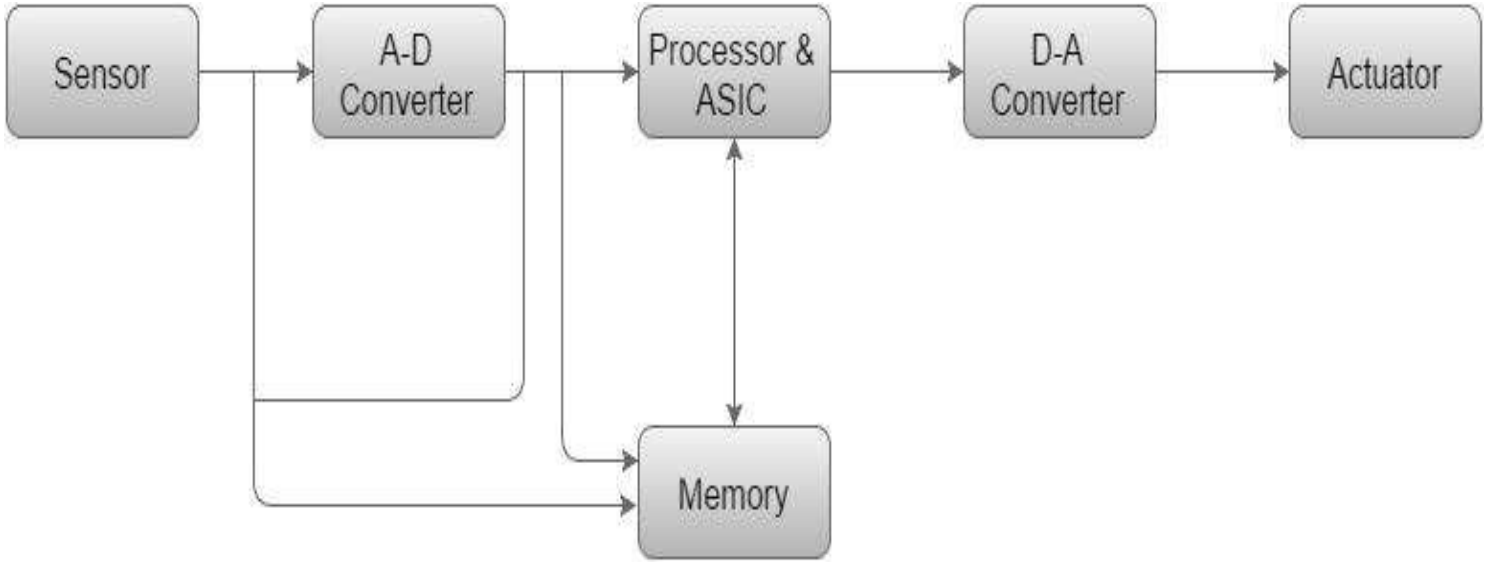
- **RTOS**

  Real Time Operating system (RTOS) supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small scale embedded system may not have RTOS.

  So these were the components of an embedded system. Let's now talk about its basic structure

Structure of an Embedded System

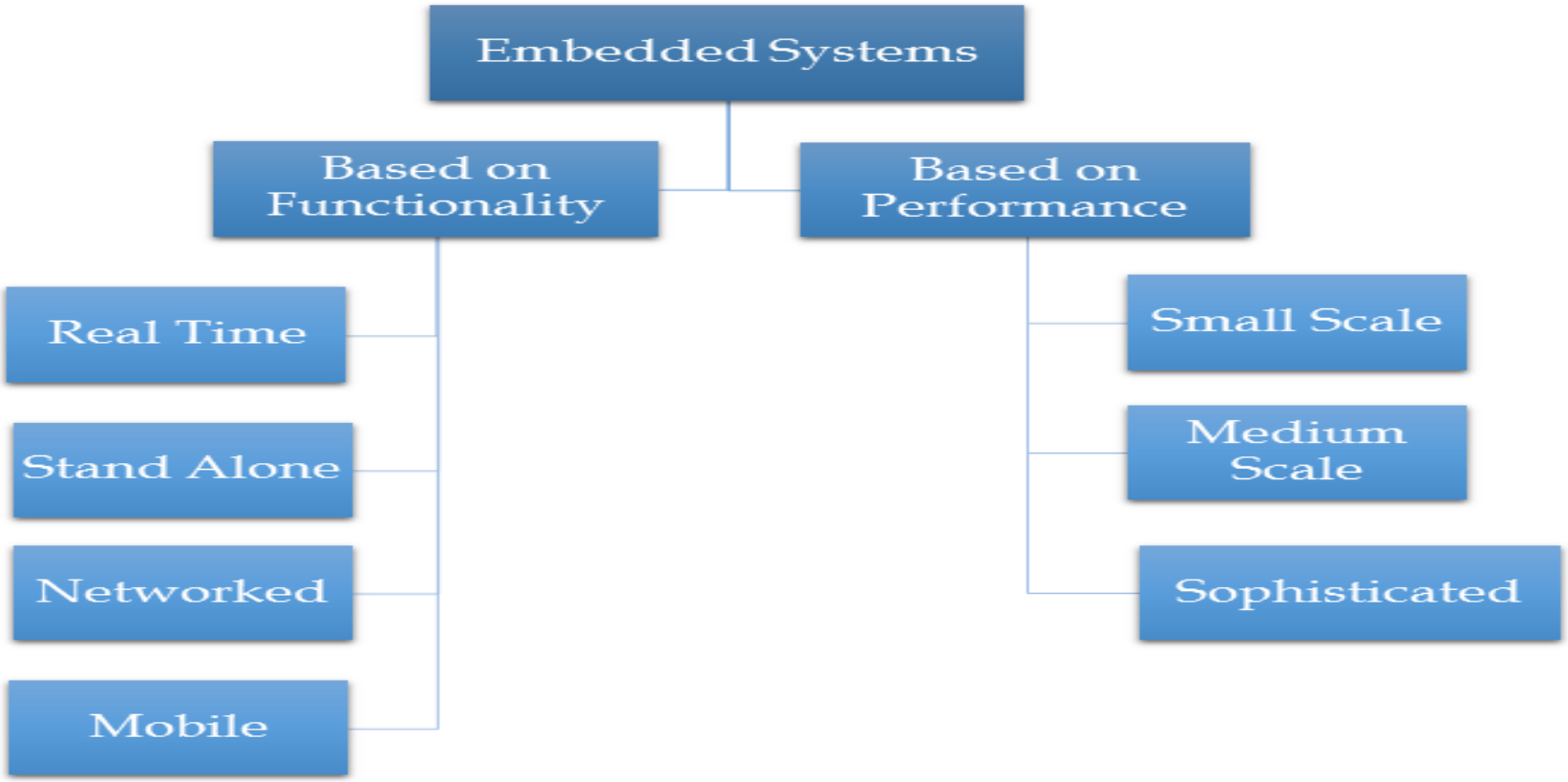
- The following illustration shows the basic structure of an embedded system –



Structure of an Embedded System

- **Sensor** − It measures the physical quantity and converts it to an electrical signal which can be read by an observer or by any electronic instrument like an A2D converter. A sensor stores the measured quantity to the memory.
- **A-D Converter** − An analog-to-digital converter converts the analog signal sent by the sensor into a digital signal.
- **Processor & ASICs** − Processors process the data to measure the output and store it to the memory.
- **D-A Converter** − A digital-to-analog converter converts the digital data fed by the processor to analog data
- **Actuator** − An actuator compares the output given by the D-A Converter to the actual (expected) output stored in it and stores the approved output.

Classification of Embedded Systems



Functionality Based Embedded Systems

Based on functions performed by embedded systems, it is divided into four types:

Real Time

An embedded system that gives an output within a specified amount of time is called a real-time embedded system. That is, in addition to a proper output, it adheres to time constraints as well. They can be further classified into soft real-time embedded systems and hard real-time embedded systems.

Stand Alone

As the name suggests, these are embedded systems that can work by themselves. In other words, they are self sufficient, and don’t require a host system or computer to function. While they will require inputs and other devices for output, the processing and work is done only by themselves. Examples include videogame consoles, music players and microwave ovens.

Functionality Based Embedded Systems

Networked Embedded Systems

- Embedded systems that are connected to a network and depend on it for their functioning are called networked embedded systems. They may or may not have smaller or less complex subsystems running to create the network. Examples include home security systems and heat sensor systems.
- These systems are connected with network that could be LAN, WAN or internet. The connection can be wireless or wired.

Mobile Embedded Systems

- Embedded systems meant for mobile communications are called mobile embedded systems. They include mobile phones, tablet computers and the like, and are usually categorized by functions like internet, calling, in addition to more complex functions seen in today's smartphones. This is the class of embedded systems which is used in portable devices.
- The examples of devices are mobile phones, cameras, music players etc.

Performance based Embedded systems

- Based on performance of microcontroller, they are divided into three types:

Small Scale Embedded Systems

- If the microcontroller used in embedded system is 8 bit or 16 bit then it is classified into small scale embedded system.
- Such systems have less complex hardware and software parts and can also be operated on batteries.
- Normally such embedded system uses Arduino boards or PIC Microcontrollers or 8051 Microcontrollers etc.

SMALL SCALE EMBEDDED SYSTEM

- Single 8 bit or 16bit Microcontroller.
- Little hardware and software complexity.
- They May even be battery operated.

- Usually “C” is used for developing these system.
- The need to limit power dissipation when system is running continuously.

Programming tools:

Editor, Assembler and Cross Assembler

MEDIUM SCALE EMBEDDED SYSTEM

- Single or few 16 or 32 bit microcontrollers or Digital Signal Processors (DSP) or Reduced Instructions Set Computers (RISC).

- Both hardware and software complexity.

Programming tools:

RTOS, Source code Engineering Tool, Simulator, Debugger and Integrated Development Environment (IDE).

SOPHISTICATED EMBEDDED SYSTEM

- Enormous hardware and software complexity
- Which may need scalable processor or configurable processor and programming logic arrays.
- Constrained by the processing speed available in their hardware units.

Programming Tools:

For these systems may not be readily available at a reasonable cost or may not be available at all. A compiler or retargetable compiler might have to be developed for this.

Performance based Embedded systems

Medium Scale Embedded Systems

- The second class is medium scale embedded system.
- It uses one or more than one 16 bit or 32 bit microcontrollers.
- It may use DSP (digital signal processor) or may use RISC (reduced instruction set computer).
- Hardware and software of these systems are complex.

Sophisticated Embedded Systems

- The third class of embedded systems is sophisticated.
- Such systems have huge hardware and software complexity.
- So they need PLA (programmable logic array), scalable or configurable processors.
- These systems have speed constraints.

Advantages of Embedded systems

- The embedded system is easy for mass production.
- The embedded system is highly reliable.
- It has very few interconnections.
- The embedded system is small in size.
- The embedded system has less expensive.
- It has fast operation.
- It has improved product quality.
- It optimizes use of system resources.
- It has low power operation.

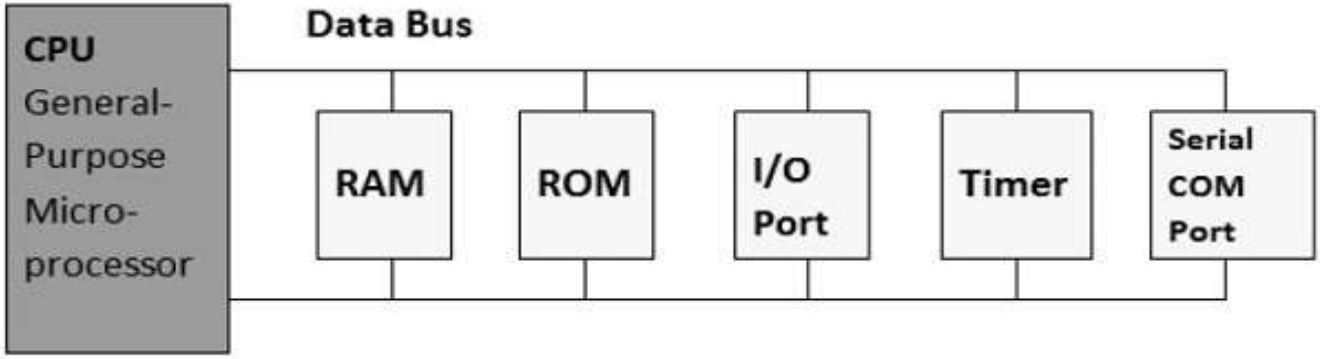
Disadvantages of Embedded systems

- The embedded systems are hard for maintenance as it is use and throw device.
- Less power supply durability if it is battery operated.
- It has hard to take backup of embedded files.

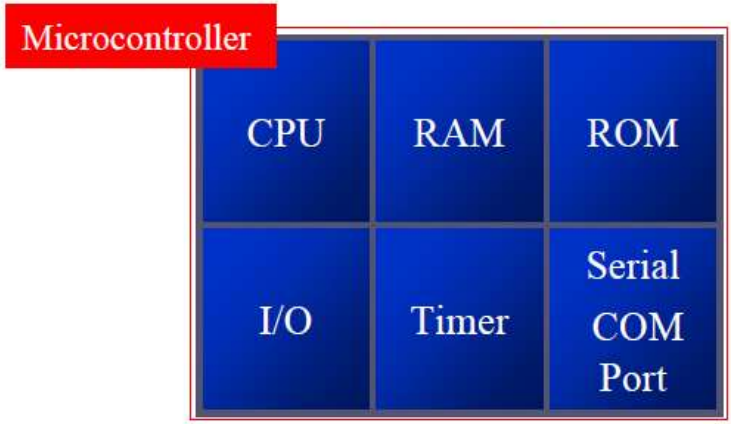
Difference between Microprocessor and Microcontroller

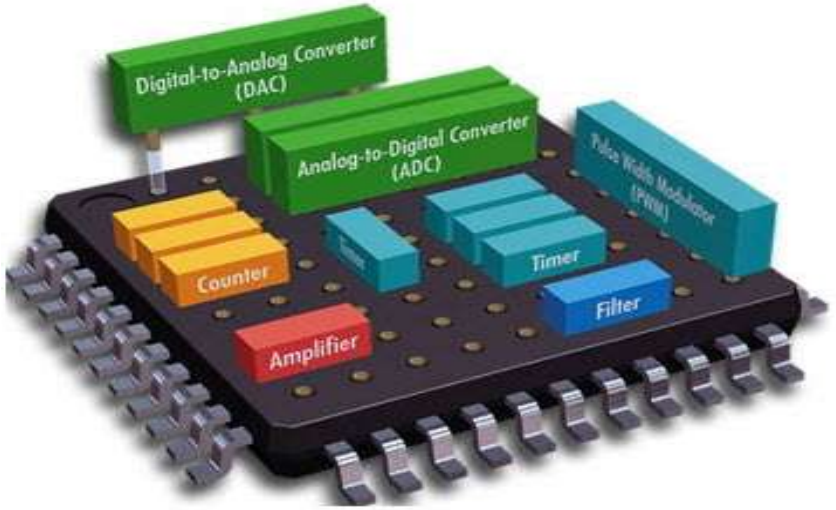
- **Microprocessor** is a general purpose device
- General-purpose microprocessors contains
- No RAM, No ROM, No I/O ports
- **Microcontroller**
- A microcontroller is a single-chip VLSI unit (also called **microcomputer**) which, although having limited computational capabilities, possesses enhanced input/output capability and a number of on-chip functional units.
- Microcontroller has CPU (microprocessor), RAM, ROM, I/O ports, Timer
- ADC and other peripherals
- Microcontrollers are particularly used in embedded systems for real-time control applications with on-chip program memory and devices.

Difference between Microprocessor and Microcontroller



A SIMPLE BLOCK DIAGRAM OF A MICROPROCESSOR





Microprocessor	Microcontroller
Microprocessors are multitasking in nature. Can perform multiple tasks at a time. For example, on computer we can play music while writing text in text editor. A PC, in contrast with the embedded system, can be used for any number of applications It has RAM memory and an operating system that loads a variety of applications into RAM and lets the CPU run them. A PC contains or is connected to various embedded products Each one peripheral has a microcontroller inside it that performs only one task	Single task oriented. For example, a washing machine is designed for washing clothes only. An embedded product uses a microprocessor (or microcontroller) to do one task and one task only There is only one application software that is typically burned into ROM

Microprocessor	Microcontroller
RAM, ROM, I/O Ports, and Timers can be added externally and can vary in numbers.	RAM, ROM, I/O Ports, and Timers cannot be added externally. These components are to be embedded together on a chip and are fixed in numbers.
Have the advantage of versatility on the amount of RAM, ROM, and I/O ports Designers can decide the number of memory or I/O ports needed.	The fixed amount of on-chip ROM, RAM, and number of I/O ports makes them ideal for many applications
External support of external memory and I/O ports makes a microprocessor-based system heavier and costlier.	Microcontrollers are lightweight and cheaper than a microprocessor.
External devices require more space and their power consumption is higher.	A microcontroller-based system consumes less power and takes less space.

Classification of Microcontroller

- **Internal bus width:**
  - 4 bit
  - 8 bit
  - 16 bit
  - 32 bit
- **Instruction set:**
  - C.I.S.C – complex instruction set computers
  - R.I.S.C – reduced instruction set computers
- **Architecture:**
  - Harvard
  - Von Neumann (or Princeton)
- **Memory:**
  - Embedded memory
  - External memomy
- **Families:**
  - ATMEL AVR - atmega, Xmega, Attinyâ€¦
  - PIC - PIC16F, PIC18F
  - ARM - ARM7, ARM9, ARM11
  - 8051 - AT89s52, p89v51rd2
  - Motorola - 68HC11

Classification According to Architectures

Harvard architecture

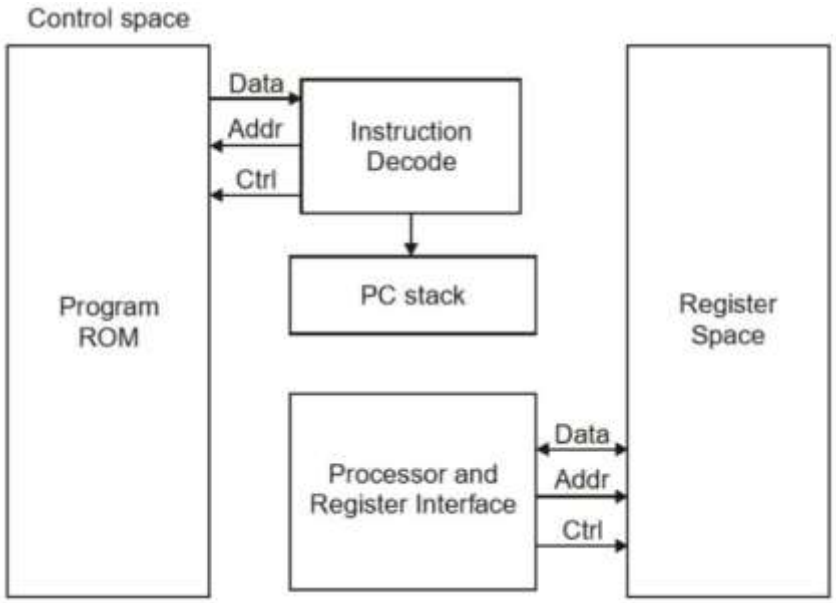
- When data and code lie in different memory blocks, then the architecture is referred as Harvard architecture.
- The Harvard architecture offers separate storage and signal buses for instructions and data. This architecture has data storage entirely contained within the CPU, and there is no access to the instruction storage as data. Computers have separate memory areas for program instructions and data using internal data buses, allowing simultaneous access to both instructions and data.

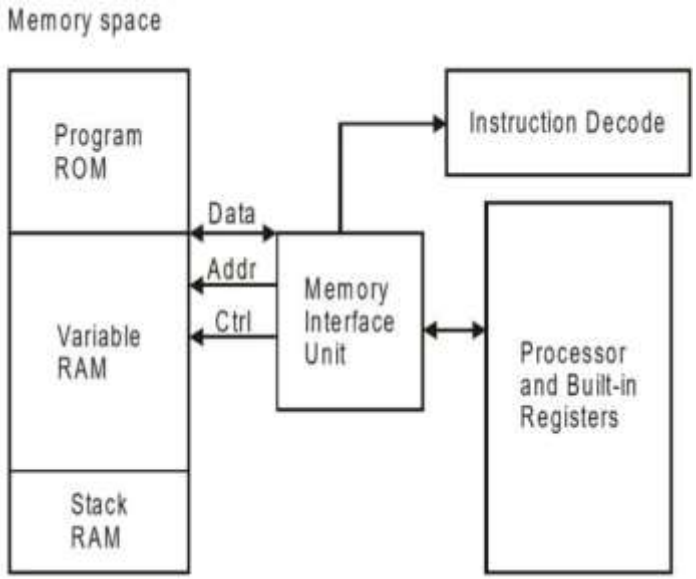
Von Neumann architecture

- In case data and code lie in the same memory block, then the architecture is referred as Von Neumann architecture. In this architecture, one data path or bus exists for both instruction and data. As a result, the CPU does one operation at a time. It either fetches an instruction from memory, or performs read/write operation on data. So an instruction fetch and a data operation cannot occur simultaneously, sharing a common bus.

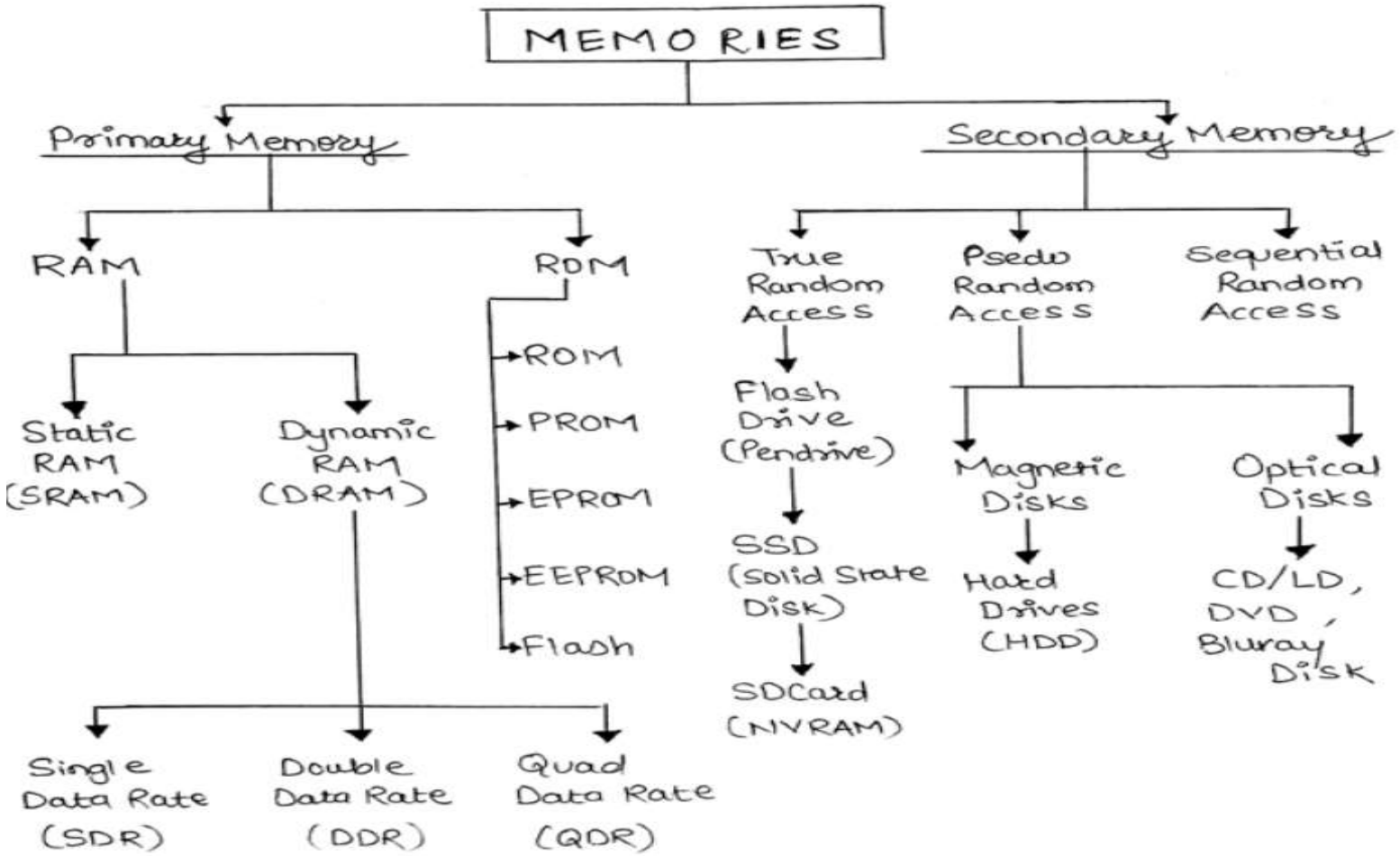
Classification According to Architectures

- Harvard
- Von Neumann





Classification of Memory



Memory classification

- Memory can be broadly classified into 2 types namely
  - Primary memory &
  - Secondary memory.
- The real difference between primary and secondary memories is **the speed/volatility tradeoffs.**

Primary memory

- Primary memory is directly addressed by the processor.
- Types
  - RAM
  - ROM

RAM: (Random Access Memory)

i) **RAM: (Random Access Memory)** Read or write memory. It stores temporary data and stack.

- Two most important types of RAM are
  - **SRAM**
  - **DRAM**.

- They are both volatile memories used as primary storage on embedded systems. But each has its place in microcontroller design. The main difference between them comes from their speed/cost tradeoffs.

Types of RAM

1. Static Random Access Memory (SRAM)

- Data is stored in the form of voltage. It is made up of flip flops. It is realized using 6 transistors. 4 transistors are part of flip flop and two transistors are for control access.
- This is the faster of the two, **approximately 4 times faster** than the DRAM.
- Since it needs more transistors per bit of data, it is also **more expensive** compared to DRAMs.

2. Dynamic Random Access Memory (DRAM)

- It stores data in the form of charge. It is made up of MOS transistors. The circuit has 1 MOSFET and a capacitor.
- The reason behind its name comes from the fact that the **data stored in this RAM needs to be refreshed every few milliseconds or else it will end up being erased**. Yes even if the power is being applied continuously the data still needs to be refreshed. This action is taken care of by a special device named **DRAM controllers.**
- The reason behind this dynamic behavior is because of the capacitor present in its design.
- Earlier the SRAM was called just RAMs but later after the introduction of DRAMs, the term “static” got introduced into its name in order to differentiate it from the DRAM technology!

	SRAM	DRAM
Construction Principle	It uses a cross-coupled flip flop configuration of transistors	It uses a capacitor transistor circuit to hold data
Cost	Relatively more expensive, it needs more transistors per bit of data it can store	Relatively less expensive, as fewer transistors per bit of storage are needed
Speed	4X more than DRAM	4X less speed
Volatility	As long as power is ON, it can store data since it uses no capacitors	Data needs to be continuously refreshed (usually in the order of 4 times a second) since the capacitors leak power.
Power consumption	Less	More
Addition components needed	None	DRAM controllers are needed to make it work like an SRAM. This controller offloads the data refreshing duties of a microprocessor and hence a DRAM coupled with a DRAM controller behaves more like an SRAM from the processor's perspective.
Application areas	Applications/scenarios that need very fast memory	Budget applications that do not need the speed of SRAMs

ROM: (Read Only Memory)

- It stores application programs from where processor fetches instruction code. It stores codes for system booting and RTOS.
- It retains content even after system is turned off. It is a non-volatile storage memory.

Types of ROM

1. MROM - Mask Read Only Memory

- MROM is the short form of Mask Read Only Memory. It is inexpensive and is the very first ROM which is hard wired device that contains a pre-programmed set of data or instructions.

2. PROM - Programmable Read Only Memory

- PROM is read-only memory chip that data can be written only once by a user. The difference between it and the read only memory is that PROM is manufactured as a blank memory, while the ROM is programmed during the manufacturing process.
- The user buys a PROM, the user will need a special device called a PROM programmer or PROM burner to write the desired data onto the blank PROM chip. The process of programming a PROM is sometimes called burning the PROM. The memory can be programmed just once after manufacturing by "blowing" the fuses, which is an irreversible process.

Types of ROM

- **EPROM - Erasable Programmable Read Only Memory**

- EPROM is a special kind of read only memory chip that has the opportunity to erase the programmed data, which the feature can be seen from its name.

- The programmable read-only memory can be programmed to write data with high voltage, and the data remains until it is exposed to ultraviolet light for lasting up to 10 minutes or longer.

- Usually, an EPROM eraser can achieve this purpose, making it possible to reprogram the memory. For this purpose, a quartz transparent window is reserved on the package of the memory for easy exposure.

Types of ROM

- **EEPROM - Electrically Erasable and Programmable Read Only Memory**
- EEPROM is also a kind of read only memory that the principle of operation is similar to EPROM which we have mentioned, but the ways to program and erase are done by exposing it to an electrical charge, so no transparent window is needed.

- It can be erased and reprogrammed about 10,000 times. Both erasing and programming take about 4 to 10 milliseconds. In the EEPROM, users can selectively erase and program any location and it can be erased one byte at a time instead of being erased the entire chip. Therefore, the process of reprogramming can be flexible but slow.

- **Flash Memory**
- Flash memory (flash) is a modern type of EEPROM. Flash memory can be erased and rewritten faster than ordinary EEPROM, and newer designs has the feature that is very high endurance (exceeding 1,000,000 cycles).

Secondary Memory

- **Secondary Memory**
- Secondary memory is not directly addressed by the processor.
- Programs and data are kept on a long term basis. It has enormous storage capacity as compared to primary memory.
- Secondary memory does not care if the power is present or not, it can happily hold its contents. But compared to Primary memory, **they are relatively slow**. Popular implementations include the Hard disks, SSDs, USB Flash drives, and SD cards.

Difference between RAM and ROM

RAM	ROM
1- Used in the computer's regular operations, after loading the OS.	Used mostly in a computer's start-up process.
2- With RAM, writing data is a fast process.	Writing data to ROM is very slow.
3- RAM is a type of volatile memory, meaning the stored data is lost when powering off.	ROM is a type of non-volatile memory, meaning that the data will not be lost when power is removed.
4- A RAM chip can store quite a lot of data, up to 16 GB.	ROM chips usually store only a few megabytes of information, around 4 MB per chip.
5- There are two main types of RAM: dynamic (DRAM) and static (SRAM).	ROM types include EPROM, EEPROM, PROM and Mask ROM.
6- Example of RAM: RAM chips like 2GB, 4GB, 8GB etc of different companies like Corsair, Kingston etc.	Example of ROM: cartridge in video game consoles, computer BIOS.

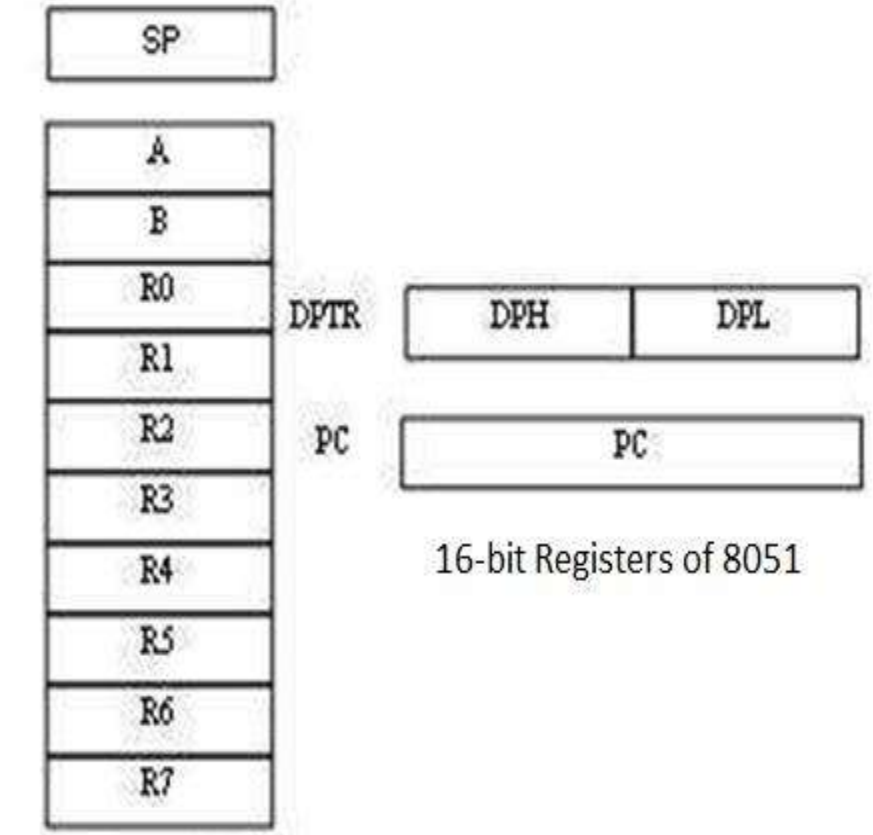
CPU

- **CPU-central processing units**
- brain of a microcontroller
- responsible for fetching the instruction, decodes it, then finally executed.
- connects every part of a microcontroller into a single system. The primary function of CPU is fetching and decoding instructions. Instruction fetched from program memory must be decoded by the CPU.

Registers of 8051

Storage Registers in 8051

- Accumulator
- R register
- B register
- Data Pointer (DPTR)
- Program Counter (PC)
- Stack Pointer (SP)



Registers of 8051 contd.

- Accumulator

  The accumulator, register A, is used for all arithmetic and logic operations. If the accumulator is not present, then every result of each calculation (addition, multiplication, shift, etc.) is to be stored into the main memory. Access to main memory is slower than access to a register like the accumulator

- The "R" Registers

  The "R" registers are a set of eight registers, namely, R0, R1 to R7. These registers function as auxiliary or temporary storage registers in many operations.

Registers of 8051 contd.

- **The "B" Register**

  The "B" register is very similar to the Accumulator in the sense that it may hold an 8-bit (1-byte) value. The "B" register is used only by two 8051 instructions: **MUL AB** and **DIV AB**. To quickly and easily multiply or divide A by another number, you may store the other number in "B" and make use of these two instructions. Apart from using MUL and DIV instructions, the "B" register is often used as yet another temporary storage register, much like a ninth R register.

- **The Data Pointer**

  The Data Pointer (DPTR) is the 8051's only user-accessible 16-bit (2-byte) register. DPTR is meant for pointing to data. It is used by the 8051 to access external memory using the address indicated by DPTR. DPTR is the only 16-bit register available and is often used to store 2-byte values.

Registers of 8051 contd.

- **The Program Counter**

  The Program Counter (PC) is a 2-byte address which tells the 8051 where the next instruction to execute can be found in the memory. PC starts at 0000h when the 8051 initializes and is incremented every time after an instruction is executed. PC is not always incremented by 1. Some instructions may require 2 or 3 bytes; in such cases, the PC will be incremented by 2 or 3.

- **The Stack Pointer (SP)**

  The Stack Pointer, like all registers except DPTR and PC, may hold an 8-bit (1-byte) value. The Stack Pointer tells the location from where the next value is to be removed from the stack. When a value is pushed onto the stack, the value of SP is incremented and then the value is stored at the resulting memory location. When a value is popped off the stack, the value is returned from the memory location indicated by SP, and then the value of SP is decremented.

Introduction to Embedded C

- Embedded C is perhaps the most popular languages among Embedded Programmers for programming Embedded Systems.

- There are many popular programming languages like Assembly, BASIC, C++, Python etc. that are often used for developing Embedded Systems but Embedded C remains popular due to its efficiency, less development time and portability.

C Programming

- C language is a structure-oriented language, developed by Dennis Ritchie. It provides less memory access using the simple compiler and delivers the data efficiently according to machine instructions. They are applicable in wide ranges from embedded systems to supercomputers.

Embedded C

- Embedded C is an extension of the C language, which is used for developing an embedded system. The syntax is similar to C language (like the main function, functions declaration, data types declaration, loops, etc). The main difference between embedded C and standard C language are input-output addressing of hardware, fixed-point operations, and processing address spaces.

C vs Embedded C

C Language	Embedded C Language
Generally, this language is used to develop desktop-based applications	Embedded C language is used to develop microcontroller-based applications.
C language is not an extension to any programming language, but a general-purpose programming language	Embedded C is an extension to the C programming language including different features such as addressing I/O, fixed-point arithmetic, multiple-memory addressing, etc.
It processes native development in nature	It processes cross development in nature
It is independent for hardware architecture	It depends on the hardware architecture of the microcontroller & other devices

C vs Embedded C

The compilers of C language depends on the operating system	Embedded C compilers are OS independent
In C language, the standard compilers are used for executing a program	In embedded C language, specific compilers are used.
The popular compilers used in this language are GCC, Borland turbo C, Intel C++, etc	The popular compilers used in this language are Keil, BiPOM Electronics & green hill
The format of C language is free-format	Its format mainly depends on the kind of microprocessor used.
Optimization of this language is normal	Optimization of this language is a high level
It is very easy to modify & read	It is not easy to modify & read
Bug fixing is easy	Bug fixing of this language is complicated

Basics of Embedded C Program

Two of the basic features of the Embedded C Program:

- Keywords
- Datatypes.

Keywords in Embedded C

- A Keyword is a special word with a special meaning to the compiler (a C Compiler for example, is a software that is used to convert program written in C to Machine Code). For example, if we take the Keil's Cx51 Compiler (a popular C Compiler for 8051 based Microcontrollers) the following are some of the keywords:

- bit-
- sbit - The **sbit** type defines a bit within a special function register (SFR).
- sfr- special function registers
- small
- large

Keywords in Embedded C

- The following table lists out all the keywords associated with the Cx51 C Compiler.

_at_	alien	bdata
bit	code	compact
data	far	idata
interrupt	large	pdata
priority	reentrant	sbit
sfr	sfr16	small
task	using	xdata

Data Types in Embedded C

- Data Types in C Programming Language (or any programming language for that matter) help us declaring variables in the program. There are many data types in C Programming Language like signed int, unsigned int, signed char, unsigned char, float, double, etc. In addition to these there few more data types in Embedded C.

- The following are the extra data types in Embedded C associated with the Keil's Cx51 Compiler.

  - bit
  - sbit
  - sfr
  - sfr16

Data Types in Embedded C

- **sbit**
- This is one kind of data type, used to access a single bit within an SFR register.
- The syntax for this data type is : sbit variable name = SFR bit ;
- Example: sbit a=P2^1;
- If we assign p2.1 as ‘a’ variable, then we can use ‘a’ instead of p2.1 anywhere in the program, which reduces the complexity of the program.

Data Types in Embedded C

- **Bit**
- This type of data type is mainly used for allowing the bit addressable memory of random access memory like 20h to 2fh.
- The syntax of this data type is : name of bit variable;
- Example: bit c;
- It is a bit series setting within a small data region that is mainly used with the help of a program to memorize something.

Data Types in Embedded C

- **SFR**
- This kind of data type is used to obtain the peripheral ports of the SFR register through an additional name. So, the declaration of all the SFR registers can be done in capital letters.
- The syntax of this data type is: SFR variable name = SFR address for SFR register;
- Example: SFR port0 = 0×80;
- If we allocate 0×80 like ‘port0’, after that we can utilize 0×80 in place of port0 wherever in the programming language to decrease the difficulty of the program.

Data Types in Embedded C

- **SFR Register**
- The SFR stands for Special Function Register. In 8051 microcontroller, it includes the RAM memory with 256 bytes, which is divided into two main elements: the first element of 128 bytes is mainly utilized for storing the data whereas the other element of 128 bytes is mainly utilized to SFR registers. All the peripheral devices such as timers, counters & I/O ports are stored within the SFR register & every element includes a single address.

Data Types in Embedded C

- The following table shows some of the data types in Cx51 Compiler along with their ranges.

Data Type	Bits (Bytes)	Range
bit	1	0 or 1 (bit addressable part of RAM)
signed int	16 (2)	-32768 to +32767
unsigned int	16 (2)	0 to 65535
signed char	8 (1)	-128 to +127
unsigned	8 (1)	0 to 255
float	32 (4)	±1.175494E-38 to ±3.402823E+38
double	32 (4)	±1.175494E-38 to ±3.402823E+38
sbit	1	0 or 1 (bit addressable part of RAM)
sfr	8 (1)	RAM Addresses (80h to FFh)
sfr16	16 (2)	0 to 65535

Different Components of an Embedded C Program

- **Steps to Build an Embedded C Program**
- There are different steps involved in designing an embedded c program like the following.
- Comments
- Directives of Processor
- Configuration of Port
- Global variables
- Core Function/Main Function
- Declaration of Variable
- The logic of the Program

Different Components of an Embedded C Program

- **Comments:** Comments are readable text that are written to help us (the reader) understand the code easily. They are ignored by the compiler and do not take up any memory in the final code (after compilation).

- There are two ways you can write comments: one is the single line comments denoted by // and the other is multiline comments denoted by /*....*/.

- In an embedded C programming language, we can place comments in our code which helps the reader to understand the code easily.

- C=a+b; /* add two variables whose value is stored in another variable C*/

Different Components of an Embedded C Program

- **Single Line Comment**
- Generally, for the programming languages, single-line comments are very useful to clarify a fraction of the program. These comments begin with a double slash (//) and it can be located anywhere within the programming language. By using this, the whole line can be ignored within a program.
- **Multi-Line Comment**
- Multi-line comments begin with a single slash (/) & an asterisk (/*) in the programming languages which explains a block of code. These types of comments can be arranged anywhere within the programming language and mainly used to ignore a whole block of code within a program.

Different Components of an Embedded C Program

- **Preprocessor Directive:**

- The lines included within the program code are called preprocessor directives which can be followed through a hash symbol (#). These lines are the preprocessor directives but not programmed statements.

- the #include directive is generally used to comprise standard libraries like study and. h is used to allow I/O functions using the library of ‘C’.
- The #define directive usually used to describe the series of variables & allocates the values by executing the process within a particular instruction like macros.

Different Components of an Embedded C Program

- A Preprocessor Directive in Embedded C is an indication to the compiler that it must look in to this file for symbols that are not defined in the program.
- In C Programming Language (also in Embedded C), Preprocessor Directives are usually represented using # symbol like #include… or #define….
- In Embedded C Programming, we usually use the preprocessor directive to indicate a header file specific to the microcontroller, which contains all the SFRs and the bits in those SFRs.
- In case of 8051, Keil Compiler has the file “reg51.h”, which must be written at the beginning of every Embedded C Program.

Different Components of an Embedded C Program

- **Configuration of Port**
- The microcontroller includes several ports where every port has different pins. These pins can be used for controlling the interfacing devices. The declaration of these pins can be done within a program with the help of keywords. The keywords in the embedded c program are standard as well as predefined like a bit, sbit, SFR which are used to state the bits & single pin within a program.
- There are certain words that are reserved for doing specific tasks. These words are known as keywords. They are standard and predefined in the Embedded C. Keywords are always written in lowercase. These keywords must be defined before writing the main program. The main functions of the keywords include the following.

Different Components of an Embedded C Program

- **Global Variables:** Global Variables, as the name suggests, are Global to the program i.e., they can be accessed anywhere in the program.

- When the variable is declared before the key function is known as the global variable. This variable can be allowed on any function within the program. The global variable's life span mainly depends on the programming until it reaches an end.

- #include<reg51.h>
  Unsigned int a, c =10;
  Main()
  {
  ……
  ……..
  }

Different Components of an Embedded C Program

- **Local Variables:** Local Variables, in contrast to Global Variables, are confined to their respective function.

- **Core Function / Main Function**

- Every C or Embedded C Program has one main function, from where the execution of the program begins . The main function is a central part while executing any program and it begins with the main function simply. Each program utilizes simply one major function since if the program includes above one major function, next the compiler will be confused in begin the execution of the program.

- #include<reg51.h>
  Main()
  {
  ……
  ……
  }

Basic Structure of an Embedded C Program (Template for Embedded C Program)

- **Main Function** . . . . . Main Function, execution begins here
  {
  Local Variables . . . . . Variables confined to main function
  Function Calls . . . . . Calling other Functions
  Infinite Loop . . . . . Like while(1) or for(;;)
  Statements . . . . .
  ....
  ....
  }

- **Function Definitions** . . . . . Defining the Functions
  {
  Local Variables . . . . . Local Variables confined to this Function
  Statements . . . . .
  ....
  ....
  }

Basic Structure of an Embedded C Program (Template for Embedded C Program)

Declaration of Variable

- The name like the variable is used for storing the values but this variable should be first declared before utilized within the program. The variable declaration states its name as well as a data type. Here, the data type is nothing but the representation of storage data. In embedded C programming, it uses four fundamental data types like integer, float, character for storing the data within the memory. The data type size, as well as range, can be defined depending on the compiler.
- The data type refers to an extensive system for declaring variables of different types like integer, character, float, etc. The embedded C software uses four data types that are used to store data in memory.
- The ‘char’ is used to store any single character; ‘int’ is used to store integer value, and ‘float’ is used to store any precision floating-point value. The size and range of different data types on a 32-bit machine are given in the following table. The size and range may vary on machines with different word sizes.
- The char/signed char data type size is 1 byte and its range is from -128 to +128
- The unsigned char data type size is 1 byte and its range is from 0 to 255
- Int/signed int data type size is 2 byte and its range is from -32768 to 32767
- Unsigned int data type size is 2 byte and its range is from 0 to 65535

Basic Structure of an Embedded C Program (Template for Embedded C Program)

- The logic of the Program

The logic of the program is a plan of the lane that appears in the theory behind & predictable outputs of actions of the program. It explains the statement otherwise theory regarding why the embedded program will work and shows the recognized effects of actions otherwise resources.

- ```
  Main
  {
  LED = 0x0f;
  delay(100);
  LED = 0x00;
  delay(100);
  }
  ```
  ```

Structure of the Embedded C program

- comments
- preprocessor directives
- global variables
- main() function

{

- local variables
- statements
- ……..
- ……..

}

- fun(1)

{

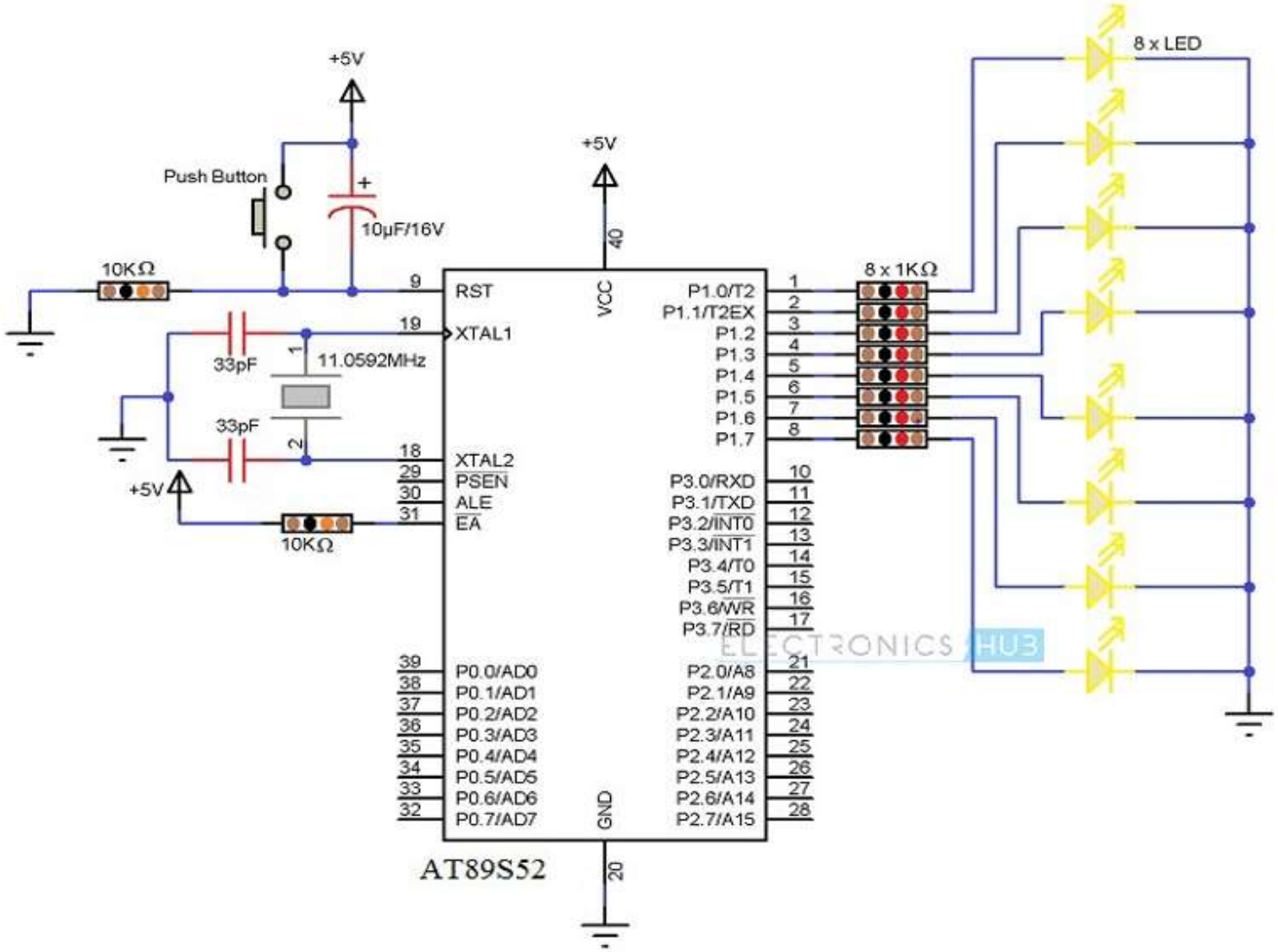
- local variables
- statements
- ……..
- ……..

}

Example of Embedded C Program

- In this example, we will use an 8051 Microcontroller to blink LEDs connected to PORT1 of the microcontroller.
- The circuit contains an 8051 based Microcontroller (AT89S52) along with its basic components (like RESET Circuit, Oscillator Circuit, etc.) and components for blinking LEDs (LEDs and Resistors).
- In order to write the Embedded C Program for the above circuit, we will use the Keil C Compiler. This compiler is a part of the Keil µVision IDE.

Circuit diagram for the example circuit.



Program

```
#include<reg51.h> // Preprocessor Directive
void delay (int); // Delay Function Declaration


void main(void) // Main Function
{
P1 = 0x00;
/* Making PORT1 pins LOW. All the LEDs are OFF.
  * (P1 is PORT1, as defined in reg51.h) */


while(1) // infinite loop
{
P1 = 0xFF; // Making PORT1 Pins HIGH i.e. LEDs are ON.
delay(1000);
/* Calling Delay function with Function parameter as 1000.
  * This will cause a delay of 1000mS i.e. 1 second */

P1 = 0x00; // Making PORT1 Pins LOW i.e. LEDs are OFF.
delay(1000);
}
}
```

Program

- void delay (int d) *// Delay Function Definition*
  {
  unsigned int i=0; *// Local Variable. Accessible only in this function.*

  / This following step is responsible for causing delay of 1000mS*
  ** (or as per the value entered while calling the delay function) */*

- for(; d>0; d–)
  {
  for(i=250; i>0; i – -);
  for(i=248; i>0; i – -);
  }
  }

Program1

1. Write a Program to toggle microcontroller port0 continuously.

```
#include<reg51.h>                /*prepocessor directive */

void main()
{

    unsigned int i;              /*local variable*/

    P0=0x00;
    while(1)
     {
     P0=oxff;                    /*statements*/
     for(i=0;i<255;i++);
     P0=0x00;
     for(i=0;i<255;i++);
     }
```

Program2

2. write a program to toggle P1.5 of 8051 microcontroller

```
#include<reg51.h>
sbit a=p1^5;

void main()
{

    unsigned int k;
    a=0x00;
    while(1)
    {
        a=0xff;
       for(i=0;i<255;i++);
        a=0x00;
       for(i=0;i<255;i++);
     }
}
```

Program 3

3. WAP to monitor P2.5 and mov the values of P2.5 to P3.6

```
#include<reg51.h>
 sbit a=P2^5;
 sbit b=P3.6;
 bit c;
 void main()
  {

      if(a==0)
      {
          b=a;
      }
   }
```

Program 4

4. WAP to Increament the values of port1 from 0 to FF

```
#include<reg51.h>
 SFR a=0x00;
 void main()
 {
      unsigned char i;
      P1=0x00;
      a=0;
      while(1)
       {
       for(i=0;i<=255;i++)
        {
          P1++;
           a++;
        }
       }
   }
```

Advantages of Embedded C

- It is very simple to understand.
- It executes a similar task continually so there is no requirement for changing hardware like additional memory otherwise storage space.
- It executes simply a single task at once
- The cost of the hardware used in the embedded c is typically so much low.
- The applications of embedded are extremely appropriate in industries.
- It takes less time to develop an application program.
- It reduces the complexity of the program.
- It is easy to verify and understand.
- It is portable from one controller to another.

Disadvantages of Embedded C

- At a time, it executes only one task but can't execute the multi-tasks
- If we change the program then need to change the hardware as well
- It supports only the hardware system.
- It has a scalability issue
- It has a restriction like limited memory otherwise compatibility of the computer.

Applications of Embedded C Program

- Embedded C programming is used in industries for different purposes
- The programming language used in the applications is speed checker on the highway, controlling of traffic lights, controlling of street lights, tracking the vehicle, artificial intelligence, home automation, and auto intensity control.